Context Sensitive Solution for Collaborative Decision Making on Quality Assurance in Software Development Processes

Oliver Kotte\textsuperscript{a,1} Aitor Elorriaga\textsuperscript{b} Dragan Stokic\textsuperscript{a} Sebastian Scholze\textsuperscript{a}
\textsuperscript{a}ATB-Bremen, Institute for Applied System Technology, Bremen, Germany
\textsuperscript{b}Innopole, Toledo, Spain

Abstract. The main objective of the paper is to create context sensitive decision support services within flexible Quality Assurance (QA) of software development projects and their resulting products. The new QA process is supported by an Internet solution composed of several knowledge, context sensitive services based on open standards that is able to detect changes in the scope and requirements of an application (or changes in its development process) and provide the adequate set of assessments as a basis for an accurate measurement of the quality of the process and product at any time and allow for effective decision making within QA. The Internet Services monitor the different stages of the software development process, interoperating with the existing applications and systems to provide quantitative information about the quality of each phase (i.e. project management, requirements gathering, functional and technical design, development and testing), the project as a whole and the resulting product. They also monitor context under which the SW is developed and decisions on QA have to be made. Data obtained in real-time by the monitoring services are used in an indistinctive way by software engineers, designers, developers, testers and managers alike for different collaborative decision making. The paper is one of the first attempts to apply context sensitivity to support decision making in QA for SW development. The approach is assessed in 2 different business cases in order to validate the results under different conditions. The first business case belongs to a large software company developing large Internet projects based on Rational Unified Process methodology. The second business case belongs to a SME developing complex projects based on agile methodologies.

Keywords. Context awareness, context modeling, quality assurance in SW development, collaborative decisions

Introduction

The objective of the paper is to create context sensitive decision support services within flexible Quality Assurance (QA) software development projects and their resulting products. The amount of information to be handled by inter- as well as intra-enterprise collaboration and the corresponding software engineering tools (document management systems, bug tracking systems, etc.) is very significant in present days, as

\textsuperscript{1} Corresponding Author.
a result of the number of ICT systems that act as information sources. The success of collaborative decision making within the software engineering process in general depends on a timely access to the relevant information by the adequate collaborator. To allow for the identification of critical changes in the software quality along its life cycle and to be able to collaboratively make decisions on optimizing the software engineering process as well as maintaining software quality it is important to know the context in which a specific monitored problem pattern occurred and under which various decisions have to be made. Tools are required to develop semantics and ontology services to filter and contextualize such information to users. The sharing of this information must be very well balanced, taking into account both the needs of the cooperative task in development and security and intellectual property rights (IPR) management issues. In addition, to be able to present to a user the knowledge that is needed in a particular situation, the system has to be aware of the context the user is currently working in, i.e. under which she/he is making decisions often in collaboration with other developers, partners, customers etc. Therefore, an approach for context awareness is needed in decision support services within QA processes.

The work presented in this paper aims at contextual adaptation and augmentation of decision processes in software development QA. The paper is structured as follows: Section 1 provides a brief overview of the state of the art with two key addressed areas: QA in SW development and context sensitivity. Section 2 presents the proposed concept of context sensitive decision support in QA, while Section 3 explains in more detail context sensitivity approach and context model. Section 4 addresses potential applications and future work.

1. State-of-the-art

1.1. Decision Making in Software QA & Control in Complex Environments

Software development has always been recognized as a complex process with unclear requirements, continuously developing technologies and advanced usage areas as well as being a highly knowledge intensive practice. Now, with the recent development towards distributed and globalized projects, this complexity increases even more. QA & Control of the software development comprises a set of activities during the whole development life-cycle to ensure the quality of the software product, developed in (geographically) distributed environments. The activities to be performed by the Quality Engineer could be developed in a strongly normalized environment, e.g. verifying fulfillment of the policies and procedures established under existing standards such as ISO 9000 series or CMMi. However these activities can also be carried out in contexts without normative burdens, but still the Quality Engineer needs to plan and execute a set of activities and organize numerous collaborative decision making processes to ensure the final quality of the software, e.g.: identifying and normalizing processes within the development and testing teams, planning and orchestrating deliverable reviews, coordinating code reviews, reaching agreements about the acceptable thresholds for the source code quality metrics, etc. Current trend is flexible and adaptive software engineering methodologies, exemplified by the strong emergence of agile methods such as SCRUM or eXtreme Programming or manufacturing methodologies shifted to the IT sphere, such as Kanban, Lean Development and others. These methods place lower emphasis on planning and
detailed specifications and more emphasis on managing and tackling changes when they occur. This makes decision processes support within QA both more complex as well as more needed.

1.2. Context Awareness, Context Aware Services, Context Modeling

Context Awareness is a concept propagated in the domains of AmI and ubiquitous computing. It is the idea that computers can be both sensitive and reactive, based on their environment. As context integrates different knowledge sources and binds knowledge to the user to guarantee that the understanding is consistent, context modeling is extensively investigated within KM research. Using context information (for context-sensitive or ubiquitous computing) is an active area of research, with various context capture methods and context languages defined. Starting with the pioneering work at XEROX PARC, other notable frameworks are Context Toolkit (Berkeley), CAMELEON project, C-OWL and the Kimura System. The current research on knowledge context is primarily oriented towards capturing and utilization of contextual data for actionable knowledge [1]. In addition, it has been shown that knowledge context could be used to classify and organize knowledge so as to realize unified management of distributed, heterogeneous knowledge in a networked enterprise [2]. However, such initiatives have specific goals, so an intense study of collaborative work and its patterns is necessary to devise a suitable context model for collaborative decision making. The overview of collaboration support systems and context-sensitive collaboration systems provided in [3] illustrates a lack in provision of knowledge context. A couple of systems to handle context awareness were proposed by the research community [4], [5], [6], [7], [8].

By context modeling, the problem of how to represent the context information can be solved. However, how to extract context from the knowledge process and how to manipulate the information to meet the requirement of knowledge enrichment remains to be solved. In the research presented in this paper it is planned to model context with ontologies, and, therefore, context extraction mainly is an issue of context reasoning and context provisioning: how to infer high level context information from low level raw context data. Based on the formal description of context information, context can be processed with contextual reasoning mechanisms [9], [10].

Defining context for applying context awareness can be difficult [11]. Informal context models are often based on proprietary representation schemes without facilities for sharing the understanding about context between different systems [12]. Existing formal context models support formality and address a certain level of context reasoning [13], [14]. The modeling of context in the case of decision support for QA in SW development process presents an additional challenge, as the services are highly dynamic and reside in distributed environments. In the case of services for monitoring, analyzing and enhancing software development processes and systems the notion of context refers to preferences and skills of users, physical capabilities of the equipment and environment conditions, coming from different kinds of information sources like bug-tracking systems, code repositories etc. Application of context awareness for SOA based software development processes and systems have not yet been sufficiently researched. With the emerging and maturing of semantic web technologies, Ontology based context modeling becomes a new trend both in academy and industry. Present research on context modeling is mostly focused on ontology. Compared to other methods, ontology based method has many advantages. Ontology allow context-
modeling at a semantic level, establishing a common understanding of terms and meaning and enabling context sharing, reasoning and reuse [15]

2. Proposed concept

Following a bottom-up approach, the SOA based Platform to support collaborative decision making for QA in SW development process is divided into 4 main blocks:

**The knowledge repositories** (including the ontology model), containing all the knowledge required by the Platform: *Knowledge on the development & QA processes* (contains all the information necessary to describe the different sub-processes of the development life-cycle: the actors, stages, outcomes, pre-requisites, IT infrastructure supporting the development project, etc.), *Knowledge on the company’s development process* (contains the company’s SW development historical data, necessary in order to obtain empirical data about acceptable thresholds for certain quality goals or to calibrate the importance of certain types of assessments so as to assign them the appropriate weights), *Knowledge on the metrics model* (contains the different metrics models that could be applied for the quantified measure of quality). *Ontology modeling* enables knowledge sharing and allows every end-user of the platform to have local ontologies with individual definitions that are compatible with a global one. Update and maintenance of the global ontology has to ensure that all participants share the same ontology, valid across different languages, namely, an inter-organizational KM system built upon distributed ontologies, locally managed and centrally integrated.

**The repository of Services** and other existing external repositories of services: This layer contains ad-hoc services (the set of new Services) or registers existing ones.

**Monitoring services** monitor the contexts of both the development life-cycle and the product. Particularly, the process monitoring services are specialized in each stage of the development process: requirements definition, design, implementation, testing and deployment. The monitoring services are interfaced to the available supporting tools present at each of the SW development stages (e.g. Requirements or Issues Management tools such as TRAC or JIRA; Wikis, repositories or Content Management Systems such as Confluence, WikiMedia or Alfresco, etc.) in order to detect any event that may occur and gather any required data from the supporting tool’s database. The monitoring services act within environment of an application (e.g. testing or productive environments) observing the events taking place at each mentioned stage and environment that could have an effect on the overall Quality Requirements of the application. Some of the events that are monitored are: publishing of requirements and design documents in internal repositories, opening, evolving or closing issues related to the project, generation of source code quality reports, deployment of an application in development, integration, production servers.

**Enhancement services** support collaborative decision making. There are different types of decisions which these services may suggest depending on the detected context e.g.: *The collection of Quality Requirements*. Depending on the events, services may propose a new scope for the Quality Requirement, a new acceptability value, may suggest decisions to deprecate existing requirements or consider creating a new one. *The set of Assessments*. These changes in the collection of Quality Requirements imply an update in the previously proposed Assessments. The services propose an updated set of specific assessment actions, which could lead to change the weights of specific
evaluations etc. The development history data can also be enhanced with data about the current development process and application. **Analytical services** analyze and provide knowledge needed for decision making: the development history data of the company in order to propose specific values for weights and thresholds of each of the concepts that will be evaluated during the assessment activities, the software metrics modeled in the knowledge base, since the services may propose a specific set of metrics which thresholds will be based on empirical analysis of the previous history data, the life-cycle and QA model in order to provide the QA Engineer with the adequate set of activities and assessments. **Reporting services** produce reports offering a quantified view of the quality of the process and product. The reports offer different abstraction levels depending on the targeted audience (development team, project management, customers). **External Services.** The platform is able to integrate already existing services for several purposes, e.g.: interfacing with specific Application Lifecycle Management (ALM) tools, providing security mechanisms, Data persistence etc. **Services Invocation Framework** for searching and making services effective: This layer acts as a proxy for the several instances of the sessions that will be open at a time. As such proxy this layer provides end users’ with the required services: the internal Services or services from external repositories. **The users’ front-end:** The platform, as an internet-based application, offers the user the possibility to obtain personalized information for a development project or deployed product through the activation of specific Services. Apart from the entry to the Services, the interface provides mechanisms to update the ontology/models, the knowledge repositories and other setup information (user management, security policy establishment, etc.), as well as aggregating other services to gather new data from the development process.

### 3. Context Sensitive Decision Support and Context Modeling

During collaboration decision making process (such as software engineering), besides the knowledge that is being created, shared and reused, attention has to be paid to the background of knowledge, the particular situation and circumstances of its occurrence, shortly its context. Context information of knowledge items and skills developed during collaborative decision making processes can be extremely valuable when the projects are finished, to extend and preserve the understanding of knowledge created and decisions made by the teams [17].

**Context Model.** The basic assumption of the proposed approach is that monitoring the development life cycle and the product enriched with context can help to be aware of any potential change that may have an effect on the Quality Requirements and influence collaborative decision making within QA. Therefore, a research key is the definition of a “holistic” and dynamic context model and ontologies to enable context awareness, allowing taking into account the context of the SW engineering approach (e.g. processes, equipment and product information, users, teams etc.) [20]. Since ontology allows for knowledge sharing, logic inference and knowledge reuse, it is a widely accepted approach for semantic-rich context modeling. Therefore, ontology is used for context modeling. Based on a context ontology, logic based context reasoning is realized such as consistency validating, subsumption checking, etc. More importantly, domain specific rules are defined to infer implicit context from explicit context, and
high level context from low level context. Other statistic and machine learning approaches can also be adopted for non-logic context reasoning.

To be usable in different domains, a general enough and also extensible context model is provided. It is in a format that meets several requirements: help to describe and capture context easily; help to manipulate context; facilitate context consumption by knowledge management services. The proposed Context Model represents an abstract description of collaborative work on software development tasks. Its concepts, attributes and relations are directly derived from the collaborative situations in dynamic industrial software engineering settings. Model is based on the collaboration patterns, i.e., typically occurring forms of collaborative work relevant for software developing tasks addressed (e.g. asynchronous/synchronous work, location, time and iterations needed in solving problems, etc.), and extended with scenario specific concepts (processes, products, technology). It was specifically needed to study and identify the patterns which characterize software process and product quality measuring and knowledge sharing in industry. The best practice collaboration/activity patterns for different collaborative decision making situations are initially defined by domain experts. By abstracting concepts and relations from these patterns/models, ontology is developed to serve as a meta-model for modeling context of different collaborative situations. This means that models of different collaborative work situations and decision making are instances of this ontology. The key ontology, entitled Activity-Centric Collaboration Ontology (ACCO), allows for representing the context of collaborative work situations in the form of explicit machine interpretable knowledge. It enables intuitive representation of knowledge about collaborative work and serves as the base for further knowledge sharing, refining and reuse. It explicitly describes collaboration related activities, people, resources, and the relationships among them. The proposed Context Model defines two models as sets of ontologies: Generic Context Model and Sector-Specific Context Model. Figure 1 presents an example of two models.

The existing context models/ontologies and extractors from different previous research [21], [22] are used as a basis for development of the context-aware solutions, but adaptations were needed to apply those solutions for the SW QA objective. The generic model is built based on re-using existing (standardized) ontologies & models from other RTD initiatives such as IntelLEO, K-NET [16] and Self-Learning [22].
Projects (for modeling of Team, Competencies, Workflows, Activities), FOAF (Friend of a Friend) describing people, the links between them and the things they create and do, SKOS (Simple Knowledge Organization System) providing a standard way to represent knowledge organization system, SWEBOK (Software Engineering Body of Knowledge). From a technical point of view it is used an OWL-based context ontology and modify/extend it using an ontology editor (Protégé) [23]. The open source semantic web framework Jena and its SPARQL query engine are used to support persistency, updating and querying on the ontology, while for reasoning existing tools such as PELLET [24] and RACER [25] are used.

**Context extraction.** Services for context sensitive enhancement of knowledge in networked enterprises have been developed in previous research projects, addressing context model to support engineers performing maintenance tasks by reusing of existing (un-)structured company knowledge [16]. This approach for context awareness and context identification is partly used as a basis for the proposed context awareness approach, but it is adapted to the industrial QA software engineering targets.

Based on the context awareness approach knowledge can be generated that is necessary to provide the envisaged decision support related to short-term operational decisions, which should be optimized regarding their effect on specific subtasks in software engineering processes, as well as long-term decisions related to the overall SW engineering process. The proposed building blocks provide awareness about situations and based on this provide an appropriate support, presenting to the user only the key values and characteristics that really matter. The context awareness concept is presented in Fig 2. Using above explained Monitoring services, data on context of SW engineering process are collected. Data are also collected from services managing social interaction (MSI services supporting communication, human resource discovery, team building etc. within collaborative decision making processes) [18]. Based on these data the actual context is detected using Context Detector services (using knowledge and context models stored in Knowledge repositories). The detected context is provided to Enhancement services to support Decision making processes. The detected context is also provided to Analytical services. On the top of that as indicated above, in outer loop using the user front end, the context model is updated based on the identified new concepts and relations.

![Figure 2: Context awareness concept](image-url)
The proposed concept follows four steps to process context-awareness as identified in [19].

4. Applications and Future Work

The proposed platform and services are currently under development and testing. The approach is assessed in 2 different business cases in order to validate the results in different conditions. The first business case belongs to a large software company developing large Internet projects based on Rational Unified Process methodology. The second business case belongs to a SME developing complex projects based on agile methodologies. The first case is characterized by high variety of SW development projects varying in scope, duration, involving geographically distributed teams where the proposed approach is likely to support decision making by providing knowledge relevant for the specific context of the current SW development project. Various collaborative decision making process are supported. One of the typical examples is negotiation on SW quality requirements and targeted value (see Figure 3). The geographically distributed experts negotiate on the relevance of various requirements where Enhancement services suggest weighting based on the identified context under which the decision process is carried out (using Monitoring services which provide needed information from various systems in the company relevant for the SW development process and Context Detector). The Analytical services analyze the situation, based on the identified context, and identify requirements/weighting used in the previous ‘similar’ project, provide them to the Enhancement services. MSI services support collaborative work (i.e. identification of experts needed for negation session, communication within the session etc.).

![Negotiation Session](image)

**Figure 3:** An example of context sensitive collaborative decision making

The main challenge in the second case, where the company is providing extensive services for the customization of its powerful SW product, is provision of knowledge from the previous customization projects which is relevant for the context of the
current project and decision making process (e.g. suggestions how the problems encountered were solved in the previous projects under 'similar' context).

In both cases software engineers, developers and testers will be able to rapidly correct inadequate trends in design, development and testing, project managers will be able to schedule deliveries with the agreed level of quality and IT managers will be able to forecast and make decisions on how to reduce the cost and increase quality of future projects. This will introduce a high level of automation in the Software Quality Management (SQM) process avoiding the traditional problems of data gathering and analysis in traditional QA.

The initial analysis and testing indicate good potentials to improved QA process by the context sensitive decision support services. The benefits are highly depended on the complexity of the development project and their dynamics. The cost/benefit ratio asks for a deeper analysis of the specific company’s development processes. The decisions which raw data are worthy to on-line collect/provide by monitoring services (which means efforts/coasts to integrate services with various systems which include these data) in order to better extract the context and support decisions making, have to be made on case basis and are specific for each company/project. The methodology on how to analyze cost/benefit ratio for various companies is developed as well. However, taking into account above mentioned trend towards flexible and adaptive software engineering methodologies, exemplified by the strong emergence of agile methods, it is likely that the proposed approach will be beneficial for many SW development companies.

The paper is one of the first attempts to apply context sensitivity to support decision making in QA for SW development. The innovative context model relevant for software engineering process is proposed and new applications of such approach for effective decision making in agile software development are elaborated. Many research problems, however, are under consideration. The key research issues to be solved are how to refine the context model to better describe dynamic software engineering tasks and decision making processes (i.e. which aspects regarding collaboration are relevant for decision making) in domain specific collaborative work, and how to integrate the context model into infrastructure and tools, in order to enhance context-sensitivity of these tools, and to facilitate context extraction from the content provided by the Monitoring services. The appropriate relations between the concepts and various associated attributes in the selected context model are currently elaborated. The privacy and IPR issues critical for the proposed approach will be solved by e.g. context sensitive definition of user access rights etc. The possibility to design and implement a prototype for a mobile interface specifically devoted to project managers is also planned.

ACKNOWLEDGMENT

This work is partly supported by the U-QASAR (Universal Quality Assurance & Control Services for Internet Applications with Volatile Requirements and Contexts) project of European Union’s 7th Framework Program, under the grant agreement no. ICT- FP7 - 318082. This document does not represent the opinion of the European Community, and the European Community is not responsible for any use that might be made of its content.
References